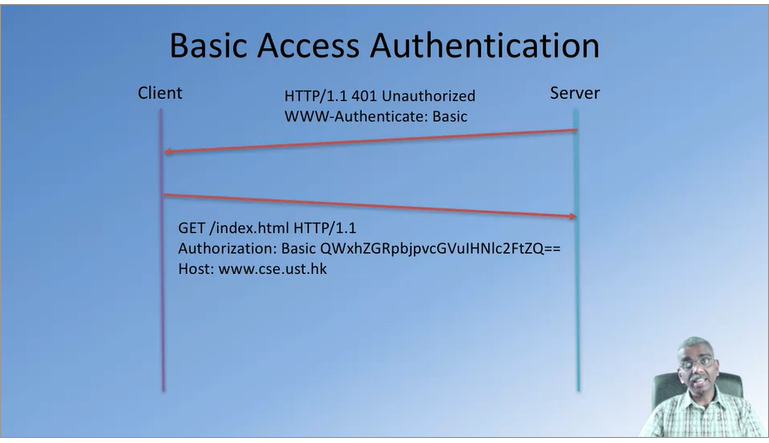
**Basic authentication**

* Security is needed to be built in the server side so not everyone can access the express server and mongoDB database

HTTP basic access authentication

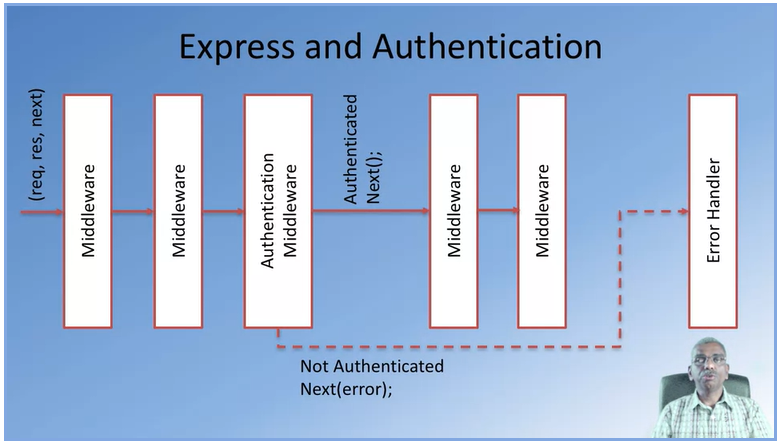
* Method for HTTP user agent to provide username and password with a request
* The server can challenge a client to authenticate itself
* Client needs to send the username and password in response



Authorization header

1. Username and password are combined into a string “username:password”
2. The resulting string literal is then encoded using Base64
3. The authorization method and a space, I.e. “basic” is then put before the encoded string

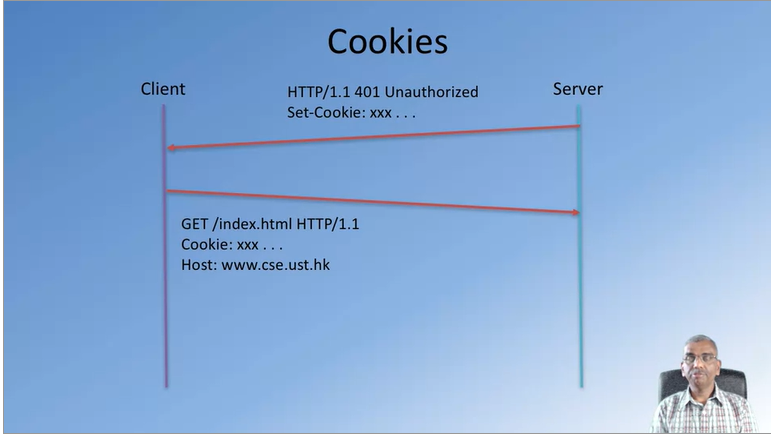
Express and authentication



**Cookies, tea and err… express sessions**

HTTP cookies

* Small piece of data sent from a web server and stored on the client side
* Each subsequent request from the client side should include the cookie in the request header



* When a client sends a request to the server side, and the client is authenticated, the web server can return a response containing a cookie
* Cookies allow the user to not have to type in their username and password every time to be authenticated

Express and cookies

* A server can set a cookie as follow in anny of the middleware
  + res.cookie(name, value, options)
* cookies are parsed in express server using the cookie-parser middleware
  + var cookieParser = require(“cookie-parser”);
* cookie-Parser parses incoming cookies and attaches them to request header
  + req.cookies.name

express and signed cookies

* signed cookie: signed w a secret key on the server side
  + digital signature w key-hash message authentication code (verifiable)
  + the secret key is a key that only the server knows, and nobody else knows
* cookie parser supports signed cookies
  + var cookieParser = require(“cookie-parser”);
  + app.use(cookieParser(“seret key”));
* parsed signed cookies made available as
  + req.signedCookies.name

express sessions

* cookies have limitations (e.g., cookies can only store a small amount of data) so express sessions can be used for functionalities that cookies cannot do
* used to track user sessions
  + combination of cookie with session id and server-side storage information indexed by session id
  + session information
    - stored by default in-memory (wiped out when server restarts)
    - stored in permanent store on server side
    - distributed session store if using multiple replicated servers: all the servers have to be able to access the session information

express-session middleware

* var session = require(“express-session”);
* var FileStore = require(“session-file-store”)(session);
* app.use(session({name: “session-id”, secret: “123 456 789”, saveUninitianlized: false, resave: false, store: new FileStore()}));
* express session information available as:
  + req.session

